Diseño de un sistema portable para la implementación de un Host USB

Sistema Host para Detección de dispositivos USB y SD

Jorge Osio*; Matias Palomeque; Jose Rapallini;
Centro de Técnicas Analógico – Digitales (CeTAD)
Facultad de Ingeniería – Universidad Nacional de La Plata
La Plata, Argentina
* Becario de Perfeccionamiento CIC – Comisión de investigaciones Científicas de la prov. de Bs. As.
Jorge.osio@ing.unlp.edu.ar; Palomequematias@hotmail.com; josrap@gmail.com

***Resumen*— El Protocolo USB se ha convertido en la interfaz universal por excelencia, no solo se encuentra disponible en las computadoras, sino también en la mayoría de los dispositivos que permiten lectura y reproducción de archivos en diversos formatos. De aquí surge la necesidad de implementar un sistema que permita detectar dispositivos USB esclavos y Memorias SD.**

Para la implementación del Sistema se ha utilizado una placa denominada "USBWiz", la cual implementa el Host y mediante diferentes comandos permite la detección de dispositivos USB y memorias SD. Dicha placa también posibilita la detección de sistemas de archivos y de una amplia variedad de dispositivos HID.

Mediante el diseño de una placa de control, se implementa una interfaz visual mediante un display LCD y un teclado que permiten recorrer el menú de opciones y entrar o salir de las mismas según se desee. Adicionalmente la placa de control contiene un sistema de encendido y apagado implementado en HW.

Mediante la interfaz visual el usuario puede detectar los dispositivos conectados, determinar el espacio libre en dichos dispositivos y acceder a los mismos para analizar su contenido.

Palabras Clave – Sistemas Embebidos, Host USB, Lenguaje de Programación C .

I. INTRODUCCIÓN

Este trabajo presenta el diseño de un sistema autónomo para la implementación de un dispositivo que permite detección y acceso a Dispositivos USB y memorias SD. El sistema se compone de 2 placas que interactúan entre sí mediante el protocolo SPI.

Una de las placas llamada USBWiz implementa el Host USB que permite la detección de dispositivos USB esclavos y la detección de memorias SD. Esta placa está formada por un Microcontrolador de la empresa NXP y un HUB USB. El Microcontrolador también contiene las librerías necesarias para la implementación de sistemas de archivos Fat32, permitiendo acceder a la información de los dispositivos USB y SD conectados.

La segunda placa ha sido diseñada para que el sistema funcione de manera autónoma, permitiendo leer datos de dispositivos USB y SD sin la necesidad de una PC. Esta placa provee la posibilidad de alimentación mediante una batería de 9V o mediante un transformador que provea entre 7,5 y 9V. Adicionalmente contiene un sistema de On-Off con seguridad para evitar errores en los dispositivos conectados. Para poder visualizar y manipular los datos se provee de un display LCD de 2x16 en donde se muestra un menú principal que permite acceder a los dispositivos y varios submenúes con la posibilidad de acceso y análisis de contenidos.

Para recorrer las distintas opciones se provee de un teclado en forma de cruz en donde la flecha superior en inferior permite recorrer las distintas opciones de un menú y la flecha derecha e izquierda permite acceder y salir de los distintos submenúes respectivamente.

Dicha placa contiene un microcontrolador atmel AT89C5131 [1], el cual se encarga de realizar el encendido y apagado del sistema, configurar el display, el teclado y de la interacción con el sistema HOST mediante el protocolo SPI. Este Dispositivo tiene la ventaja de programación en circuito mediante una interfaz USB y es programado en lenguaje C [2], para lo cual se utiliza el entorno de desarrollo Keil [3], lo que facilita mucho la actualización del Firmware.

II. DESCRIPCIÓN DE PLACA USBWIZ

El USBwiz chipset [4], permite acceder a archivos FAT (12, 16 o 32) sobre una SD y 2 dispositivos USB simultáneamente. USBwiz tiene 3 cores FAT independientes y una amplia variedad de comandos. Además, USBwiz soporta varios dispositivos usb, tales como teclados, joysticks, impresoras, celulares, ftdi, prolific, etc [5]. Se puede acceder a estos dispositivos a través de comandos simples enviados sobre un bus UART, SPI o I^2C.

La placa USBwiz-OEM, que se muestra en la Figura 1, contiene un chipset USBwiz y otros bloques adicionales.

Se encuentran disponibles las librerías de código fuente 'C' para usar USBwiz gratuitamente. Las librerías se comunican con el USBwiz a través de un simple driver.

La ventaja de usbwiz es que permite manejar 3 dispositivos a la vez sin la necesidad de un buffer. La desventaja es que no permite "long file name", es decir que no soporta nombres de archivos mayores a 8 caracteres.

Figura 1. Placa USBwiz

El chip usado para implementar el host usb es el ISP1160, esto posibilita soportar 2 dispositivos usb al mismo tiempo. El integrado provee 2 puertos usb host con un hub interno.

El microcontrolador que contiene los drivers usb y permiten detectar dispositivos usb es el lpc2134 [6], cuyo diagrama en bloques se muestra en la figura 2.

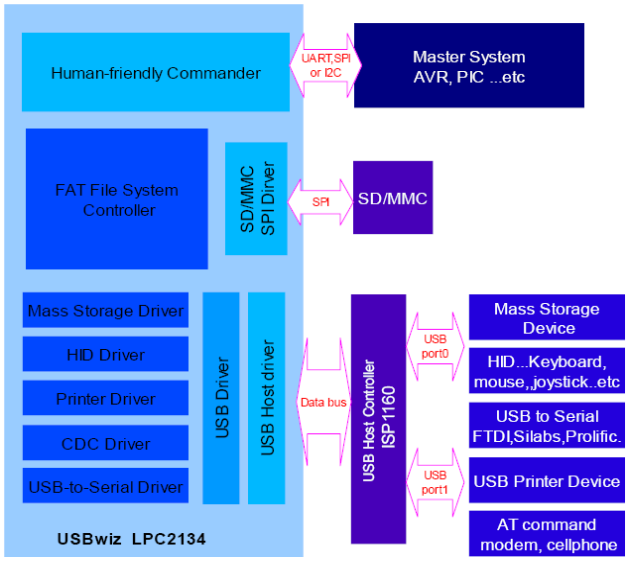

Figura 2. Diagrama en bloques del microcontrolador LPC2134

El LPC2134 es un microcontrolador basado en un CPU ARM7TDMI-S de 32/16 bits, con emulación en tiempo real, que combina el microcontrolador con la memoria flash embebida de 256KB de alta velocidad. Una interfaz de memoria de 128 bit de ancho y una Arquitectura que habilita ejecución de código de 32 bits a la frecuencia máxima de clock [6].

La interfaz con esta placa se puede realizar mediante los 3 protocolos antes mencionados. Para controlar esta placa se dispone de una serie de comandos, en donde los más significativos se muestran en la tabla siguiente.

TABLA I. COMANDOS DE CONTROL USBWIZ [4]

Nombre	Comandos de control
	Descripción
FM	Monta el sistema de archivos
II	Proporciona información del Dispositivo
IL	Inicializa la lista de archivos y capetas
NF	Proporciona el nombre del siguiente archivo o carpeta
OF	Abre un archivo para lectura o escritura
CD	Permite cambiar de directorio
CF	Cierra un archivo
RW	Lee un archivo y escribe en otro archivo
MS	Proporciona información de tamaño del dispositivo y espacio libre

III. DISEÑO DE LA PLACA DE CONTROL Y ALIMENTACIÓN

Para lograr un sistema portable se diseñó una placa que provee alimentación al sistema, ya sea, por medio de una batería de 9V o mediante alimentación externa entre 7,5 y 9V. En la figura siguiente se muestra el esquemático del circuito de alimentación y de on-off de la placa diseñada.

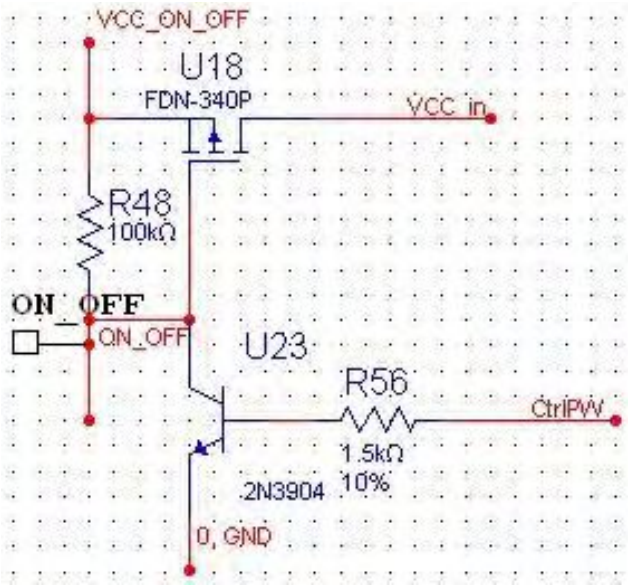

Figura 3. Circuito On-Off

La alimentación llega un regulador de 5V, que se utiliza para generar la tensión necesaria para alimentar los dispositivos USB conectados. Por otro lado la placa USBwiz provee alimentación de 3,3V para alimentar el resto del Sistema.

En la Figura 4 se muestra la placa diseñada, la cual se encarga de realizar el control, proveer alimentación, la interfaz a un display LCD y a un teclado. Se debe destacar que el esquemático y el Layout de la placa fueron diseñados mediante el software multisim [7], muy potente para dichas aplicaciones.

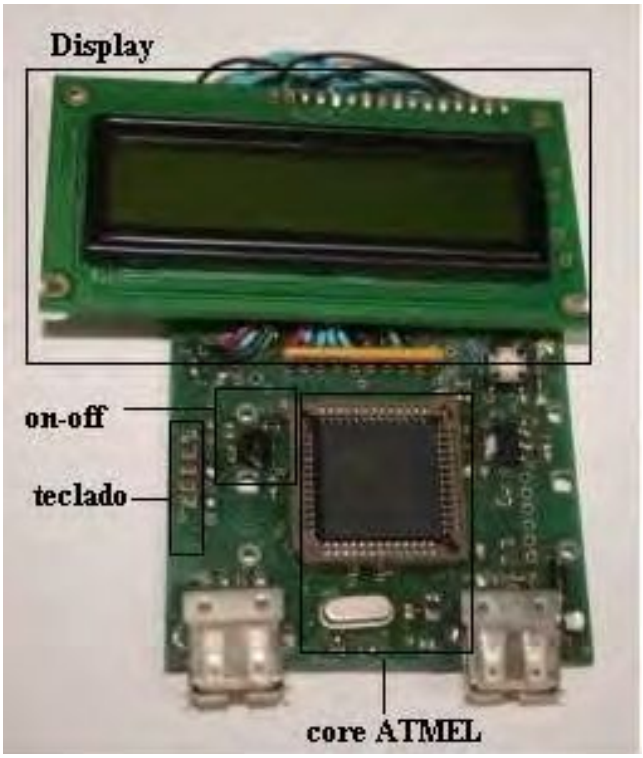

Figura 4. Placa de control e interfaz humana

Esta placa se comunica mediante el protocolo SPI a la placa USBwiz encargada de detectar los dispositivos USB y SD.

El core de esta placa es un Microcontrolador ATMEL AT89C5131 de 8 bits que permite el encendido y apagado del dispositivo mediante un pin que indica que la placa debe encenderse y con un retardo de 2 segundos que permite el apagado del sistema.

La figura 5 muestra el diagrama en bloques del microcontrolador, en donde se observa un módulo SPI utilizado para la comunicación con la placa USBwiz, un módulo usb utilizado para la actualización del firmware mediante programación en circuito. Módulos de temporización que permiten generar los diferentes retardos necesarios para la correcta funcionalidad del sistema y los puertos de Entrada/Salida que permiten la interfaz con el teclado y con el display LCD [8].

Por otro lado se debe destacar la gran cantidad de bloques de memoria de dicho dispositivo, lo que permite implementar un programa con una gran variedad de funcionalidades y una amplia cantidad de opciones que conforman el menú del sistema completo.

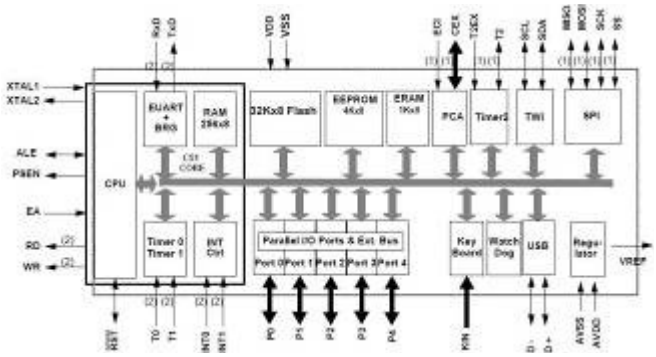

Figura 5. Diagrama en Bloques del Microcontrolador ATMEL

IV. DESCRIPCIÓN FUNCIONAL DEL SISTEMA

En la Figura 6 se muestra un diagrama en bloques del Sistema completo, en donde fácilmente se lo puede dividir en cinco bloques principales:

- ***Circuito on-off:*** Por un lado se encuentra el circuito de on-off y el software correspondiente que permite su funcionamiento.
- ***Interfaz con la placa USBwiz:*** Está formada por el módulo SPI y por un conjunto de funciones que implementan los comandos necesarios.
- ***Menú de opciones:*** Implementado por software.
- ***Configuración de Teclado:*** El teclado se configura por software y su estado sirve como entrada al menú de opciones.
- ***Configuración de Display:*** Se configura por software y permite visualizar el menú.

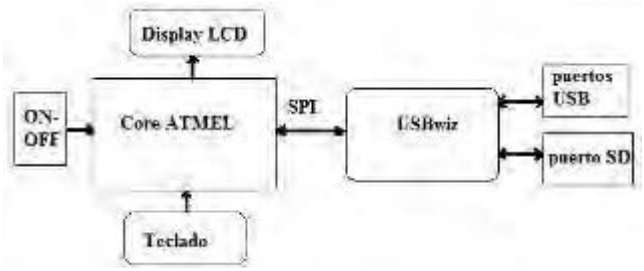

Figura 6. Diagrama en Bloques del Sistema completo

A. *Circuito ON-OFF*

Como se muestra en el esquemático de la Figura 3, el circuito de encendido y apagado está formado básicamente por 2 transistores, al presionar el botón de encendido se pone a tierra la línea ON-OFF activando el transistor mosfet, lo que permite alimentar el resto del circuito. Una vez alimentado el microcontrolador, energiza mediante un pin la línea CtrlPW que se conecta directamente con la base del transistor bipolar, polarizándolo. Mientas se mantenga esta condición el transistor mosfet permitirá que el circuito se mantenga energizado.

Al presionar el pulsador de apagado, la línea intPW se pone a tierra indicando al microcontrolador que se debe cerrar toda transferencia antes de poner en cero la línea CtrlPW, la cual desenergizará la base del transistor bipolar. De esta manera el

gate el mosfet quedará en alta impedancia, haciendo que la tensión VCC_in caiga a cero.

B. Interfaz con la placa USBwiz

La interfaz entre la placa de control y la placa USBwiz se realiza mediante el protocolo SPI a una frecuencia de clock 8Mhz. El Microcontrolador ATMEL posee un módulo SPI, el cual se configura mediante una función en C, la cual es llamada cada vez que se desea enviar comandos desde la placa de control hacia la USBwiz. A continuación se describe el código necesario para enviar un caracter a la USBwiz.

```
void PutC(int8 ch)
{
        while(BUSY_PIN);
        if( ch==NDT)
        {
                // se necesita enviar HDT
                SSEL_PIN=0;
                SPDAT = HDT;
                while ( SPSTA != 0x80);
                ProcessRX(SPDAT);
                SSEL_PIN=1;
                ch=0;// se envía cero para indicar 0xFF
        }else if( ch==HDT )
        {
                // se debe enviar HDT
                SSEL_PIN=0;
                SPDAT = HDT;
                while ( SPSTA != 0x80);
                ProcessRX(SPDAT);
                SSEL_PIN=1;
                // se envía HDT por segnda vez para indicar 0xFE
        }
        SSEL_PIN=0;
        SPDAT = ch;
        while ( SPSTA != 0x80 );
        ProcessRX(SPDAT);
        SSEL_PIN=1;
    }
```

C. Menú de opciones

El menú de opciones está formado por máquinas de estados. Por defecto, el sistema se encuentra recorriendo el menú principal que muestra los 3 dispositivos posibles como se observa en la Figura 7. Una vez detectado un dispositivo, se accederá al primer submenú que permite ver la memoria del dispositivo o acceder al mismo mediante otra máquina de estados como muestra la Figura 8.

Figura 7. Menú principal – selección de dispositivos

Por último, dentro del dispositivo se accede a un menú dinámico que permite recorrer los contenidos hasta los niveles más anidados.

Dentro de un dispositivo, sea cual sea el nivel en que se encuentre, si se deja presionada la flecha derecha se vuelve al menú principal del dispositivo.

(a) Primer submenú – Lectura de memoria de la SD

(b) Primer submenú – Lectura del sistema de archivos

8. Primer submenú. (a) lectura de memoria. (b) lectura de archivos

A continuación se muestra la estructura de programa que ejecuta la máquina de estados que permite acceder al primer subnivel.

```
switch(level_screen){
case 0: {//nivel 0 o top level, muestra los dispositivos accesibles

        switch(screen0){
                case 0:{      //Device 1
                     break;          }
                case 0x01: {  …..                    // Device 2}
                case 0x02: {  …..                    // Device 3 }
          }
case 0x01: {   //nivel 1, permite el acceso a tamaño de memoria o archivos
                  switch(screen1){
                  case 0:{      //Memory u1
                              switch(date_task){
                              case 0:{
                                   aux1= MountFileSystem(USB _PORT_1);
                                    if(aux1!=0) {
                                      lcd_line(0x40,"No Device     ");
                                      work_flag=0;}
                                   else
                                     date_task=0x01;
                                    break; }
                                case 0x01:{
                                             aux1=GetC();
                                  if(aux1=='\r')
                                             date_task=0x02;
                                   break;  }
                                 case 0x02: {
                                  aux1=GetC();
                                  if(aux1=='\r')
                                        date_task=0x03
                                         break;}
                                   case 0x03: {
                                   lcd_line(0x40,"~~ Memory U1 1/2");
                                  devise='1';
                                  work_flag=0;
                                  date_task=0;
                                   break;}
                        }
                  case 0x01:// read u1
                  case 0x02: //memory u2
                ……}
```

```
Case 0x02 {//menu dinámico break;}
```

D. *Configuración de Teclado*

El teclado está formado por 2 pulsadores para recorrer un mismo nivel en el menú, uno para cambiar de nivel y otro para volver al estado anterior. Adicionalmente, se proveen 2 pulsadores para el encendido y apagado del dispositivo.

Todos los pulsadores poseen resistencias de pull-up, lo que quiere decir que al presionarlos se pondrán a tierra las respectivas líneas.

La configuración por software consiste en una máquina de estados, en donde los estados son "new" (cuando se presiona una nueva tecla), "running" (contiene una serie de tiempos de espera, que sirven para interpretar diferentes funcionalidades de algunos pulsadores), "stopped" (permite analizar cuál es el pulsador que cambió), "delayed" (se analiza cual fue la tecla presionada más de 2 seg).

De acuerdo a la información obtenida en estas funciones, se envía el mensaje correspondiente al display.

E. *Configuración de Display*

El display LCD se conecta al puerto 0 mediante un array de resistencias de pull-up, esto es así puesto que el puerto 0 del ATMEL de un puerto de alta impedancia. Por otro lado, mediante un potenciómetro se regula la tensión de referencia VEE para regular la luz de dígito [8].

La configuración por software consiste en una función de inicialización, que realiza los pasos necesarios para poder entablar una comunicación con el display utilizando 8 líneas de datos y una función que es llamada cada vez que se desea envía un string de caracteres al display.

Lo primero que hace el dispositivo es inicializar el LCD mediante la siguiente función:

```
void init_lcd(void){                // Rutina de inicio
        LCD_BL=0;                           //backlight on
        task_lcd_in[read_pt_lcd].state=new;
        DelayMs(200);
        lcd_putcomm(0x38);  // Modo 2 líneas, 5x7 dots y modo 8 bits
        DelayUs(80);
        lcd_putcomm(0x0c);  // Display on, cursor off y blink off
        DelayUs(80);
        inicio_clear();  // Clear display     con retardos (solo para el inicio)
        DelayMs(4);
        lcd_putcomm(0x06);   // Increment mode, entire shift off
   DelayUs(80);
        init_message();
 }
```

Luego pone el mensaje inicial, correspondiente a la selección de los 3 dispositivos posibles.

La función principal del display llamada "LCD()" también consiste en una máquina de estados formada por los estados: "new" (en donde se verifican punteros del LCD y cmd mediante el seteo del estado de entrada y salida de la pila), "running" (se muestra el mensaje en pantalla verificando la posición del cursor, el estado y cmd), "stopped" (estado que indica que se terminó de mostrar el mensaje), "reset_msg" (resetea el puntero, limpia el mensaje y entra nuevamente en new para mostrar un nuevo mensaje).

El display recibe los mensajes desde el menú, de acuerdo al pulsador presionado del teclado.

V. RESULTADO OBTENIDOS

Luego de realizado el programa y de las pruebas de HW se comprobó que el programa se volvía inestable cuando se intentaba acceder a un puerto en donde no había dispositivo conectado. Luego de varias pruebas se comprobó que no se había contemplado la posibilidad de volver al menú principal en el caso de no tener un dispositivo conectado.

Otro problema que dejaba al dispositivo en un estado desconocido se daba cuando se insertaba detectar una memoria SD, en donde el dispositivo se bloqueaba intentando acceder a la misma. Haciendo pruebas de bajó nivel se comprobó que el mensaje que devuelve la SD al realizar el montaje de la misma es diferente al de los dispositivos USB y eso provoca que el sistema entre en un estado de bloqueo.

Solucionados estos dos grandes inconvenientes de software se procedió a realizar las pruebas de hardware correspondiente mediante el prototipo que se muestra en la Figura 9.

En las pruebas de Hardware se comprobó que el sistema es bastante lento, por un lado por el hecho de ser un dispositivo USB 1.1 full speed. Pero, por otro lado comparando con un dispositivo usb 1.1 conectado a la PC se comprobó un retardo un poco mayor, el cual se debe a la configuración interna del dispositivo USBwiz [4].

En la Tabla 2 se puede observar la comparación entre una PC con USB 1.1 y el sistema host implementado. Evidentemente el tamaño del archivo influye mucho en el tiempo de demora entre un sistema y el otro. Esto se verifica porque para copiar un archivo de 1MB el dispositivo host tarda el doble que la PC y para copiar uno de 23MB el dispositivo host tarda 6 veces más. Lo que nos lleva a suponer que el método de transferencia de datos utilizado por la placa USBwiz es muy ineficiente si se utiliza con dispositivos USB full-speed [5].

TABLA II. TIEMPOS ENTRE EL HOST USB DISEÑADO Y UNA PC CON USB1.1

Tiempos		**Pruebas Realizadas**
PC	**Host USB**	
-	35 s	Lectura de memoria de un dispositivo de 15Gigas
12 s	16s	Lectura de memoria de un dispositivo de 1Giga
15 s	37 s	Archivo de 1.1MB de un dispositivo USB a otro
2m	12 m	Archivo de 23MB de un dispositivo USB a otro

Por otro lado las pruebas con la SD fueron de igual manera exitosas, permitiendo leer el espacio disponible de memoria y acceder al contenido de la misma como en los dispositivos USB.

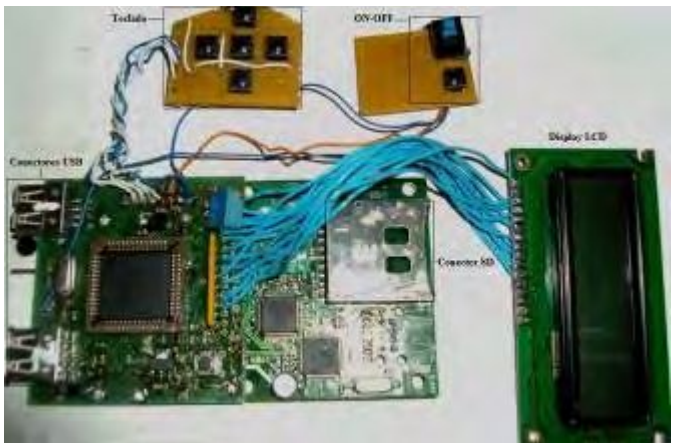

Figura 9. Prototipo del Sistema Host USB Portable.

VI. CONCLUSIONES

A grandes rasgos se ha cumplido con los objetivos del proyecto, mediante la obtención de un dispositivo totalmente portátil, con la posibilidad de alimentación con batería o alimentación externa.

Se logró implementar un Sistema que permite acceder a dispositivos USB y SD, posibilitando el acceso a la información de cada dispositivo.

A pesas que este dispositivo se encuentra en etapa de test, se ha logrado un dispositivo funcional, estable en cada uno de los minués y confiable.

Este dispositivo tiene las limitaciones de la placa USBwiz que soporta USB full speed y tiene un modo ineficiente de transferir información, limitando la velocidad de funcionamiento a velocidades inferiores a 12Mbits por segundo.

VII. TRABAJO A FUTURO

Como trabajo a futuro se pretende unificar el sistema implementando en un mismo Microcontrolador y de esta manera reducir notablemente el tamaño del dispositivo y optimizar los recursos utilizados. La Parte de implementación del Host USB ya se ha implementado en el trabajo "*Sistema Host USB para gestionar archivos entre dispositivos*" mediante el Microcontrolador LPC2478. Por lo que solo resta adaptar el código diseñado para el dispositivo ATMEL a dicho microcontrolador y de esta forma se obtendrá un dispositivo portable Funcional completamente diseñado por el Laboratorio.

Por otro lado, se desea implementar el USB 2.0 que soporta transferencia a High speed y mejorar el sistema de archivos implementando el sistema fat fs, lo que posibilitará la lectura y escritura de archivos de nombre superior a 8 caracteres y ampliar el menú de opciones, con la posibilidad de crear archivos de texto e ingresar texto a los mismos.

REFERENCIAS

[1] User´s manual: Atmel USB microontrollers AT89C5131.

[2] Brian W. Kernighan, Dennis M. Ritchie. " El lenguaje de programación C" 2 Edición.

[3] User´s manual: Entorno de desarrollo KEIL uVision 3. Antonio Moreno Fernández-Caparrós.

[4] User´s manual: "USBwiz User´s manual" rev 2.27 April 20, 2009.

[5] Jan Axelson " USB Mass Storage, Designing and Programming Devices and Embedded Hosts"

[6] User´s Manual: Philips. "LPC 213x. microcontrollers" rev 02 , 25 July 2006.

[7] Addlink software científico. " Curso de Multisim 9" año 2006.

[8] User´s Guide: INTECH " Liquid Cristal Display Module"